ATARI® PROGRAM EXCHANGE

The Soft Warehouse

CALCULUS DEMON™

Symbolic partial derivatives and
indefinite integrals of expressions (ages 16 and up)

Cassette: 32K (APX-10155) Diskette: 40K (APX-20155)

User-Written Software for ATARI Home Computers

The Soft Warehouse

CALCULUS DEMON™

Symbolic partial derivatives and
indefinite integrals of expressions (ages 16 and up)

Cassette: 32K (APX-10155) Diskette: 40K (APX-20155)

Distributed By

The ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

To request an APX Product Catalog, write to the address above, or call toll-free:

800/538-1862 (outside California)
800/672-1850 (within California)

Or call our Sales number, 408/727-5603

Trademarks of Atari

Printed in U.S.A

TABLE OF CONTENTS

INTRODUCTION

OVERVIEW

CALCULUS DEMON, the third program in a series that includes ALGICALC and POLYCALC, provides a comprehensive tool for automatically deriving symbolic partial derivatives and indefinite integrals of expressions. Individually, the programs offer a quick way to perform various kinds of operations in symbolic algebra and calculus. Collectively, they constitute a complete computational package.

CALCULUS DEMON is best at calculus, although it does contain some algebraic capabilities. It also offers some trigonometric, logarithmic, and exponential simplification. The algebraic expressions CALCULUS DEMON can handle include trigonometric, inverse trigonometric, logarithmic, and exponential functions. The program also provides various automatic and optional mathematical transformations to simplify results or aid integration. Unlike most programming languages, which can evaluate an expression only if all variables have numeric values, CALCULUS DEMON can do true non-numeric operations.

REQUIRED ACCESSORIES

ATARI BASIC Language Cartridge

Cassette version

32K RAM
ATARI 410 Program Recorder

Diskette version

40K RAM
ATARI 810 Disk Drive.

CONTACTING THE AUTHOR

Users wishing to contact the author about CALCULUS DEMON may write David Stoutemyer at:

The Soft Warehouse
P.O. Box 11174
Honolulu, Hawaii 96828

USING CALCULUS DEMON

ENTERING PROBLEMS

It's easy to use CALCULUS DEMON for solving problems. You type in expressions, using the conventions and notation described in the help topic and in this manual. If you make a typing mistake, use the DELETE/BACK S key to erase the error and type in the correction. Once you've entered the problem accurately, you press the RETURN key to signal the program to process your problem. CALCULUS DEMON beeps when it completes the operation and displays the results. This feature lets you turn your attention elsewhere when CALCULUS DEMON is doing time-consuming operations, since it notifies you audibly when you can enter another problem.

INVALID ENTRIES

If your entry is invalid or the solution (or transformation) exceeds memory capacity, a ^ displays below the character where processing could not proceed, and an appropriate error message appears. For example:

```
+-----------------------------------+
|? X + Y/Z)^ 2                      |
|          ^                        |
|                                   |
|ERROR, expected different character|
|Enter a problem, or enter another  |
|question mark for help             |
|                                   |
|?_                                 |
+-----------------------------------+
```

Most of the error messages are quite specific and detailed. They usually make the cause obvious -- at least after you obtain a little experience with the program. Section 6, ALPHABETIZED ERROR MESSAGES: PROBABLE CAUSES, contains an alphabetized list of the err messages together with likely causes.

INTERRUPTING COMPUTATIONS

To interrupt a computation, press the SYSTEM RESET key and refer to the program's loading instructions. Since this process is time-consuming, make sure that you carefully enter the desired expansions.

THE KINDS OF OPERATIONS CALCULUS DEMON CAN PERFORM

CALCULUS DEMON can derive symbolic partial derivatives and indefinite integrals of expressions. These expressions can be algebraic expressions, including trigonometric, inverse trigonometric, logarithmic, and exponential functions. As an example, the program can evaluate the partial derivative

$$\frac{\partial}{\partial x} \text{ sin log } (ax\text{^}2 + b)$$

giving

$$2ax(ax^2+b)^{-1}\ \cos\ \log\ (ax^2+b)$$

CALCULUS DEMON can determine the indefinite integral

$$\int x^2\ (ax^3 + b)^{1/3}\ dx$$

giving

$$.25\ a\ (ax^3 + b)^{4/3}$$

The program also provides various automatic and optional mathematical transformations in order to simplify results or aid integration. For example, optional tranformations permit simplification of an expression such as

$$(\sin x + \cos x)^2 - 1$$

into the equivalent expression

$$\sin (2x)$$

USING CALCULUS DEMON

ENTERING PROBLEMS

It's easy to use CALCULUS DEMON for solving problems. You type in expressions, using the conventions and notation described in the help topic and in this manual. If you make a typing mistake, use the DELETE/BACK S key to erase the error and type in the correction. Once you've entered the problem accurately, you press the RETURN key to signal the program to process your problem. CALCULUS DEMON beeps when it completes the operation and displays the results. This feature lets you turn your attention elsewhere when CALCULUS DEMON is doing time-consuming operations, since it notifies you audibly when you can enter another problem.

INVALID ENTRIES

If your entry is invalid or the solution (or transformation) exceeds memory capacity, a ^ displays below the character where processing could not proceed, and an appropriate error message appears. For example:

```
+------------------------------------+
|? X + Y/Z)^ 2                       |
|         ^                          |
|                                    |
|ERROR, expected different character|
|Enter a problem, or enter another   |
|question mark for help              |
|                                    |
|?_                                  |
+------------------------------------+
```

Most of the error messages are quite specific and detailed. They usually make the cause obvious — at least after you obtain a little experience with the program. Section 6, ALPHABETIZED ERROR MESSAGES: PROBABLE CAUSES, contains an alphabetized list of the err messages together with likely causes.

INTERRUPTING COMPUTATIONS

To interrupt a computation, press the SYSTEM RESET key and refer to the program's loading instructions. Since this process is time-consuming, make sure that you carefully enter the desired expansions.

THE KINDS OF OPERATIONS CALCULUS DEMON CAN PERFORM

CALCULUS DEMON can derive symbolic partial derivatives and indefinite integrals of expressions. These expressions can be algebraic expressions, including trigonometric, inverse trigonometric, logarithmic, and exponential functions. As an example, the program can evaluate the partial derivative

$$\frac{\partial}{\partial x} \text{ sin log (ax^2 + b)}$$

giving

$$2ax(ax^2+b)^{-1} \cos \log (ax^2+b)$$

CALCULUS DEMON can determine the indefinite integral

$$\int x^2 \, (ax^3 + b)^{1/3} \, dx$$

giving

$$.25 \ a \ (ax^3 + b)^{4/3}$$

The program also provides various automatic and optional mathematical transformations in order to simplify results or aid integration. For example, optional tranformations permit simplification of an expression such as

$$(\sin x + \cos x)^2 - 1$$

into the equivalent expression

$$\sin (2x)$$

THE HELP TOPICS

To familiarize yourself with CALCULUS DEMON's features before working on your own problems, you can request any of eleven instruction screens. Each HELP topic explains a convention CALCULUS DEMON follows or a notation it uses. The HELP screens also display one or two problems you can type in for practice. Each topic presumes you are familiar with the material covered in the preceding topics, so it's a good idea to work through the HELP screens in order. After you read an explanation, try the practice problems or make up problems of your own. You can also return to the HELP menu at any time by typing a question mark and pressing the RETURN key. Or, you can go directy to another topic by typing the lesson number and a question mark and then pressing the RETURN key.

The eleven topics displayed on the HELP menu are as follows:

```
+-----------------------------------------+
|Help is available on                      |
|                                          |
|1: Expressions         7: Compound Lines  |
|2: Algebraic Expand    8: Quoting         |
|3: Functions           9: Re-evaluation   |
|4: More Expansion     10: Derivative      |
|5: Assignments        11: Integral        |
|6: Non-display                            |
|                                          |
|Enter a topic number followed by a        |
|question mark                             |
|                                          |
|Study topics in order the first time      |
|                                          |
|See manual for details                    |
|?_                                        |
+-----------------------------------------+
```

To select a topic, type its corresponding number and a question mark, and then press the RETURN key. For example, to select Expressions, type:

```
1? [RETURN]
```

The instruction screen looks like this:

```
+-----------------------------------------+
|After each question mark prompt, enter   |
|an expression using digits, decimal      |
|points, parenthesis and 1-letter         |
|variables together with operators +, -   |
|/, * and ^.  The latter means raising    |
|to a power.  *, meaning multiply,can     |
|be omitted.  Following each such input   |
|an equivalent expression in diskplayed,  |
|derived by reordering, collecting        |
|similar terms, and other mild            |
|transformations.  As examples, type      |
|                                         |
|3 X^2 Y + 5 (XYX - 7)                    |
|(A^2*B)^.5 / A                           |
|                                         |
|After entering several such examples,    |
|enter a question mark to see the help    |
|menu.                                    |
|                                         |
|?_                                       |
+-----------------------------------------+
```

To try the practice problems, type one in and press the **RETURN** key. After a pause, **CALCULUS DEMON** will display the results and you can verify your understanding of the lesson.

EXPRESSIONS

Expressions are formed following standard algebraic notation, and supplemented with traditional computer input conventions permitting formula entry on one line.

Variables

All letters are automatically converted to upper case. For example, the input "b + B" will generate the response "2 B".

Constants

Constants consist of adjacent digits, with an optional decimal point imbedded or at either end. Examples are 82, 82., .82 and 0.820. Even a decimal point alone is interpreted as the constant zero, but it is clearer to type 0.

Operators

Between two subexpressions, use + for addition, - for subtraction, / for division, ^ for raising to a power, and * for multiplication. The latter can be omitted if you prefer. For example, 5.2*37*X*4 and 5.2 37X4 both indicate the product of four quantities 5.2, 37, X, and 4.

Signs

As in ordinary math notation:

1. - can also be used to indicate negation when preceding an expression.

2. Although unnecessary, + can also be used as a sign preceding an expression,

For example, 8/+--+-4 is equivalent to -2.

Blanks

Blanks are allowed anywhere except within a constant or within a function name (e.g., SIN or COS).

Blanks are not required except when you want to multiply adjacent constants (e.g., 8*3.2 can be entered as 8 3.2). Also, certain sequences of three immmediately adjacent letters denote built-in function names. Therefore, it is a good habit to include blanks between adjacent variables to be multiplied. (e.g.,enter A*J*K as A J K instead of as AJK).

Order of operations

When no parentheses indicate otherwise, CALCULUS DEMON processes algebraic operators in the following order:

first: ^
+ and - when used as signs preceding subexpressions
* and /
last: + and - when used for addition and subtraction.

For example, CALCULUS DEMON evaluates the expression -2^2 as -(2^2), which is -4, rather than as (-2)^2, which is +4.

Multiple exponentiation

When no parentheses indicate otherwise, CALCULUS DEMON evaluates successive exponentiations right-to-left (<--). For example, 2^3^2 means 2^(3^2) which is 2^9 or 512, rather than (2^3)^2 which is 8^2 or 64.

Multiple multiplication and division

When no parentheses indicate otherwise, CALCULUS DEMON evaluates successive multiplication and divisions left-to-right (-->). For example, 6/3*2 means (6/3)*2 which is 4, rather than 6/(3*2) which is 1.

Multiple addition and subtraction

When no parentheses indicate otherwise, CALCULUS DEMON evaluates successive additions or subtractions left-to-right (-->). For example, 6-3+2 means (6-3)+2 which is 5, rather than 6-(3+2) which is 1.

Indicating division

With CALCULUS DEMON you use the operator / to indicate division in a one-line format, rather than building a fraction on more than one line. Therefore, you must use parentheses to get the desired effect when numerators or denominators have more than one term. For example, you can enter

$$\frac{X + 3}{X + 7}$$

as (X+3)/(X+7), but not as X+3/X+7, which would be interpreted as

$$X + \frac{3}{X} + 7$$

or to enter it as (X+3)/X+7, which would be interpreted as

$$\frac{X+3}{X} + 7$$

It would also be inappropriate to enter this example as X+3/(X+7), which would be interpreted as

$$X + \frac{3}{X+7}$$

Similarly, you must use parentheses for denominators of more than one factor unless you use repeated division to get the desired effect. For example, you can enter the fraction

$$\frac{3\,X}{5\,Y}$$

as 3X/(5Y) or else as 3X/5/Y using repeated division. In contrast, the entry 3X/5Y would be interpreted as (3X/5)Y, which is

$$\frac{3\,X\,Y}{5}$$

Scientific notation

You can use multiplication together with exponentiation to enter a number in scientific notation. For example, you can enter $8.7\ 10^{11}$ as 8.7 10^11. Since most integers this large aren't exactly representable in Atari BASIC, the resulting display is 8.7E+11. In this context, BASIC uses the letter E to represent "*10^" where numerous zeros would otherwise be required to denote the number. However, you can't use this E-notation to

enter problems for the DEMON, because the notation conflicts with the convention of implied multiplication when using the letter E as a variable. In output, the context and tight spacing make it possible to distinguish E-notation from the variable E. However, you may wish to avoid using the variable E, to avoid any possible confusion.

Complex numbers

The DEMON does not have any special facility for treating complex numbers. Although subexpressions such as (-1)^.5 can be accommodated, they are not fully simplified according to the conventions of complex arithmetic.

Arithmetic Limitations

Arithmetic is done using the appropriate arithmetic provided by BASIC. Consequently, you may notice roundoff error in the coefficients and exponents of results. For example, a number that should have been 2.5 may appear as 2.444444389.

The Atari BASIC arithmetic also replaces numbers having magnitude less than about 10^{-98} by zero.

Numbers with magnitude exceeding about 10^{96} cause one of the two error messages

ERROR, expected smaller magnitude

or

ERROR, expected easier problem

ALGEBRAIC EXPAND

The help message for algebraic expansion is:

```
+-----------------------------------------+
|Type < to turn on expansion of           |
|positive integer powers of sums and      |
|full distribution of numerator or        |
|denominator factors over numerator or    |
|sums.  Type > to turn off these          |
|transformations.  For example, type      |
|                                         |
|<(X+Y)^2>                                |
|<(X+Y) (X-Y)                             |
|(X+Y)/(X-Y) + 5>                         |
|                                         |
|After entering several such examples,    |
|enter a question mark to see the help    |
|menu.                                    |
|                                         |
|?_                                       |
+-----------------------------------------+
```

These three examples respectively expand to

```
2 X Y + X^2 + Y^2,

    X^2 - Y^2,

(X-Y)^-1 X + (X-Y)^-1 Y + 5.
```

As illustrated by the last example, expressions are not put over a common denominator. Compared to the companion program ALGICALC, the DEMON is algebraically weakest with respect to the lack of factoring.

Note also that negative integer powers of sums and products of negative powers of sums are not expanded. Compare, for example, the input and response

```
? <(X+1)^-2>
(X+1)^2
```

with the input and response

```
?<1/(x+1)^2>
(x^2+2x+1)^-1
```

Therefore, alternative input forms can be used to control expansion for resulting negative powers.

Without reduction over a common denominator followed by factoring, expansion of negative powers tends to defeat the pattern-driven integration. This is the reason for discriminating against expansion for negative powers.

It is easy to forget whether expansion is on or off, and this is a frequent cause of confusion, delay, exhaustion of storage, or failure to discover an antiderivative. However you can always insure that algebraic expansion is set, as desired, by beginning a line with < when you want algebraic expansion or beginning a line with > when you want no such expansion.

FUNCTIONS

The help message for functions is:

```
+------------------------------------------+
|Expressions can include the six radian    |
|trig functions SIN, COS, TAN, COT, SEC    |
|and CSC, along with their first three     |
|inverses ASN, ACS and ACT.  Expressions   |
|can also contain the natural log and      |
|exponential functions, spelled LOG and    |
|EXP.  Parentheses can be omitted from     |
|around arguments that are constants,      |
|variables, functional forms, or their     |
|negatives.  Automatic transformations     |
|exploit symmetry, inversion, logs of      |
|powers, and numeric arguments.  As        |
|examples try                              |
|                                          |
|> COS -X                                  |
|EXP LOG X                                 |
|LOG (X^2)                                 |
|4 ATN 1                                   |
|                                          |
|After entering several such examples,     |
|enter a question mark to see the help     |
|menu.                                     |
|                                          |
|?_                                        |
+------------------------------------------+
```

These four examples respectively transform to

COS X

X

2 LOG X

3.14159267

As indicated, parentheses can be omitted from around simple arguments of functions. In this regard, note that when no parentheses indicate otherwise, functions are applied before exponentiation. For example, LOG X^2 means (LOG X)^2 rather than LOG(X^2).

MORE EXPANSION

The help message for more expansion deals with the effects of both "algebraic" and "harmonic" expansion upon functional forms:

```
+------------------------------------------+
|When ALGEBRAIC expansion is turned on     |
|via <, logs of products or quotients      |
|are rephrased as sums or differences      |
|of logs, and other trig functions are     |
|replaced by sines and cosines.  Typing    |
|[ turns on HARMONIC expansion, which      |
|transforms positive integer powers and    |
|products of sines and cosines into        |
|'linear combinations' of sines and        |
|cosines of multiple angles and angle      |
|sums and differences.  Typing ] turns     |
|off harmonic expansion.  As examples try  |
|                                          |
|< LOG (XY/Z)                              |
|TAN X CSC X                               |
|[ SIN X COS Y                             |
|SINX^2 + COSX^2 ]>                        |
|                                          |
|After entering several such examples,     |
|enter a question mark to see the help     |
|menu.                                     |
|                                          |
|?_                                        |
+------------------------------------------+
```

These four examples respectively transform to:

LOG X + LOG Y - LOG Z

COS X ^-1

0.5 SIN (X-Y) + 0.5 SIN (X+Y)

1

Harmonic expansion is so-named because it results in a representation of any sinusoidal polynomial as a linear combination of its harmonics. In other words, any polynomial in sines and cosines is transformed into an equivalent polynomial that is linear in sines and cosines. For example:

```
? <[ 8 SIN X ^3 COS (3X-Y) ])
-SIN(6X-Y) + 3 SIN(4X-Y) - 3 SIN(2X-Y) - SIN Y
```

This transformation is provided because it greatly aids integration, but the transformation is also useful for Fourier Analysis in which a waveform is decomposed into its harmonics. Also, this transformation may simplify trigonometric expressions, as illustrated by the above help message example SINX^2 + COSX^2.

ASSIGNMENTS

Before tackling assignments, be sure that you have practiced enough to have a good understanding of the concepts in the previous sections.

The help message for assignments is:

```
+----------------------------------------+
|An entry of the form                    |
|                                        |
| variable = expression                  |
|                                        |
|causes the resulting value of the       |
|expression to be ASSIGNED to the        |
|variable for use in subsequent          |
|expressions.  For example try typing    |
|                                        |
|P = 5(X+Y+Z)                            |
|P + 1/P + X LOG P                       |
|                                        |
|After entering several such examples,   |
|enter a question mark to see the help   |
|menu.                                   |
|                                        |
|?_                                      |
+----------------------------------------+
```

The program displays the assigned value the same as if only the expression were entered. Thus, these two examples result in the corresponding expressions

5 X + 5 Y + 5 Z

LOG (5 X + 5 Y + 5 Z) X + (5 X + 5 Y + 5 Z)^-1 + 5 X + 5 Y + 5 Z

Naming and saving a result for use in expressions is often useful for the following reasons:

1. Fitting an entire expression in a single entry may be impossible otherwise.

2. Correctly entering a sequence of short formulas may be easier than entering an equivalent lengthy formula.

3. Repetitious entry may be avoided by naming and saving a subexpression that occurs several places within an expression or that is to be used in several different expressions.

Initially each of the 26 variables has its name as its value. As described in previous sections, each expression is simplified immediately after you enter it. Whenever a variable is encountered in an input expression during this simplification process, the variable contributes the value it has that time. For CALCULUS DEMON, this process of simplification combined with replacement of variables by their current values is called

evaluation.

After the expression on the right side of an assignment is evaluated, the resulting value replaces the former value of the variable on the left side.

Examples:

Suppose that at the beginning of a CALCULUS DEMON dialogue when the value of every variable is its name, we enter the assignment

```
? Y = X + 3
```

causing the new value of Y to display as:

```
X + 3
```

Next, suppose we enter the assignment

```
? Z = <Y^2>
```

causing the new value of Z to display as:

```
X^2 + 6 X + 9
```

Now for the first subtle point: suppose we next enter the assignment

```
? Y = C + 5
```

causing the new stored value of Y to display as:

```
C + 5
```

If we then enter the expression

```
? Z
```

The resulting display is still

```
X^2 + 6 X + 9
```

despite the changed value of Y and the fact that Z was assigned a value that originally depended upon Y. The reason for this effect is that after contributing the value X+3 to the expression Y^2, all traces of Y have disappeared from that result X^2+6X+9, which was assigned to Z. For efficiency, no record is kept of the fact that Z received its value from the evaluation of Y^2. Consequently, subsequent changes to Y cannot have an effect on Z.

This phenomenon should not surprise anyone familiar with a programming language that relies upon assignments. However, the interaction of assigned and unassigned variables exhibits a related but more surprising aspect. Continuing our session still further, suppose that we enter the assignment

```
? X = W-1
```

causing the display

```
W - 1
```

If we then enter the expression

```
? Z
```

the resulting display is still

```
X^2 + 6 X + 9
```

the same as before, despite the changed value of X. In contrast to no trace of Y, X is explicitly present in the value that was assigned to Z. However, whenever a variable receives a new value, the program does not automatically update other stored values that contain the name of the variable.

This stable-expression mechanism is fast and simple to implement. Moreover, it is sufficiently general to permit stepwise formation of any expression. We must simply enter the assignments in bottom-up order so that assignments of subexpressions to intermediate variables precede use of those variables in the final expression or in other intermediate expressions. For example, if the objective of the above session was to form Z in terms of W, then this objective could be accomplished by the sequence:

```
? X = W-1
W - 1

? Y = X+3
W + 2

? Z = <Y^2>
W^2 + 4 W + 4
```

These examples illustrate this principle:

> Despite the use of an equal sign, a sequence of assignments should be regarded as a sequence of updates to memory locations rather than as a set of simultaneous or independent equations.

This illustration shows why only a variable is allowed on the left side of an assignment. It would be less confusing to use a different character for assignment, but for the sake of familiarity to BASIC and FORTRAN programmers, CALCULUS DEMON follows their use of the equal sign for assignment.

You can assign a variable an expression containing the same variable. For example:

```
? Q = Q + 1
Q +1
```

```
? (Q^2)
Q^2 + 2Q + 1
```

Although the result is what should be expected based on the above discussion, the result may be surprising at first. Consequently, you may prefer to avoid the practice as much as possible by introducing additional variable names to achieve the intended results. However, assignments containing the same variable on both sides are useful for clearing a variable.

Assignments containing the same variable on both sides are also useful for increasing the numeric value of a variable in order to evaluate a formula for a sequence of values. For example, using "..." to represent computations involving K:

```
? K = 1
1

 ...

? K = K +1
2

 ...

? K = K + 1
3
```

and so on.

If you don't intend to use an algebraic or a numeric result in subsequent expressions, then there is no reason to waste memory space by saving the result as the value of a variable. In such cases, it is easier and wiser to enter the desired expression.

NON-DISPLAY

The help message for non-display is

```
+-----------------------------------------+
|Follow an assignment with a semicolon    |
|to suppress display of an assigned       |
|value.  For example, try                 |
|                                         |
|P = S+1;                                 |
|P^3                                      |
|                                         |
|After entering several such examples,    |
|enter a question mark to see the help    |
|menu.                                    |
|                                         |
|?_                                       |
+-----------------------------------------+
```

The above two lines result in the line

(S + 1)^3

In sequences of assignments, there are often intermediate assignments for which the resulting display is of no interest. In such cases it may be desirable to suppress the displays avoid their distraction and to postpone scrolling off the top of the screen. Consequently, the program suppresses display of any expression or assignment followed by a semicolon.

COMPOUND LINES

The help message for compound lines is:

```
+----------------------------------------+
|Several assignments can precede an      |
|assignment or expression in a single    |
|entry, separated by semicolons.  For    |
|example, try                            |
|                                        |
|P=R+1; Q=L+P; Q^2                       |
|                                        |
|After entering several such examples    |
|enter a question mark to see the help   |
|menu.                                   |
|                                        |
|?_                                      |
+----------------------------------------+
```

This example produces the single result:

(L + R + 1)^2

Placing several assignments on a line can help postpone scrolling displayed results off the screen. Also, this permits you to initiate a lengthy sequence of calculations and then ignore the screen while the calculations occur.

QUOTING

The help message for quoting is:

```
+----------------------------------------+
|To prevent a name from contributing     |
|its value to an expression, precede     |
|the name by an apostrophe.  This        |
|provides a way of clearing the value    |
|of a variable back to its name.  For    |
|example, try                            |
|                                        |
|P=L+5; P^2                              |
|P='P; P^2                               |
|                                        |
|After entering several such examples,   |
|enter a question mark to see the help   |
|menu.                                   |
|                                        |
|?_                                      |
+----------------------------------------+
```

The results of these two examples are:

(L + 5)^2

P^2

After assigning a value to a variable it is often desired to assign the variable its own name, which was the original value. This feature saves precious memory space and it clears the variable, permitting once again using it as an unencumbered name. For example, sometime after assigning X the value W-1, it may be desired to clear the value of X bal to its own name. You might imagine that this could be accomplished with the assignment

? X = X

However, as previously mentioned, all variables in an input expression ordinarily are evaluated. Consequently, the right side of the above assignment would evaluate to W-1, which then would be reassigned to X. What we want here is to suppress evaluation of a variable, and preceding the variable by an apostrophe accomplishes this objective. If we enter

? X = 'X

then the displayed new value of X is the name X, as desired.

This quote operator is applicable only to variables, and the precedence of this operator exceeds all others.

Besides the above use, it is occasionally useful to apply a quote operator to variables within a more complicated expression. This use of the quote operator has the effect of clearing a variable locally within an expression without disturbing the assigned value outside the expression. For example, suppose that X has the value of W-1 that we wish to preserve, but that as a side calculation we wish to expand (X+1)^5 without having anything substituted for X. We could do so by merely entering the expression

? <('X + 1) ^5>.

RE-EVALUATION

Before tackling re-evaluation, be sure you have practiced and thoroughly understand expansions, assignments, and quoting.

The help message for re-evaluation is:

```
+----------------------------------------+
|If we enter the line                    |
|                                        |
|H='H; P=H+1; H=2; P^2                   |
|                                        |
|then H becomes 2 after P becomes H+1,   |
|so the displayed result is (H+1)^2      |
|rather than 9.  However, we can follow  |
|the expression P^2 with the operator @  |
|to RE-EVALUATE P^2 so that H therein    |
|is updated to its new value 2:          |
|H='H; P=H+1; H=2; P^2 @                 |
|                                        |
|@ has the same precedence and left-to-  |
|right order as addition                 |
|                                        |
|After entering several such examples,   |
|enter a question mark to see the help   |
|menu.                                   |
|                                        |
|?_                                      |
+----------------------------------------+
```

Note that @ is used somewhat differently in the companion APX program ALGICALC.

As another example of using @, suppose that for each of the values y=x+1, ..., y=x+10 we wish to expand the expression

$$2y^5 - y^4 + 7y^3 + 12y^2 - 13y + 863$$

We could do this in the bottom-up fashion as follows:

```
?Y=X+1; 2Y^5-Y^4+7Y^3 +12Y^2-13Y+863
2X^5 + 9X^4 + 23X^3 + 47X^2 + 38X + 870
? Y=X+2; 2Y^5 - Y^4 + 7Y^3 + 12Y^2 - 13Y + 863
2X^5 -Y^4 + 79X^3 + 190X^2 + 247X + 989

...

? Y=X+10; 2Y^5 - Y^4 + 7Y^3 + 12Y^2 - 13Y + 863 >
2X^5 + 99X^4 + 1967X^3 + 19622X^2 + 98327X + 198933
```

However, repetitious entry of the lengthy polynomial is tiresome. This example is better suited to a top-down approach wherein we first assign the lengthy polynomial to another variable such as Z, then we successively re-evaluate Z with Y assigned X+1, then X+2, and so on:

```
? < Z = 2Y^5 - Y^4 + 7Y^3 + 12Y^2 - 13Y + 863;

Y=X+1; Z@
2X^5 + 9X^4 + 23X^3 47X^2 + 38X + 870

? Y=X+2; Z@
2X^5 + 19X^4 + 79X^3 + 190X^2 + 247X + 989

...

? Y=X+10; Z@>
2X^5 + 99X^4 + 1967X^3 + 19622X^2 + 98327X + 198933
```

Much less typing is involved for this approach. Moreover, some users often prefer to work in such a top-down manner, starting with an overall "summary" value containing names, substituting a subexpressions for some of the names in this value. Depending upon the particular expressions involved, the bottom-up, the top-down, or mixed approach can involve less typing, memory consumption or computation time. Experiment with both extremes and various compromises to gain experience for the most effective approach.

It is important to note that each re-evaluation is only one level deep. For example, if the values of B, C and D are their own names, then the entered sequence

```
? A=B+1; B=C+2; C=D+5; A@
```

yields the display

```
C + 3
```

whereas entering

```
? A @ @
```

yields

```
D + 8
```

The ability to re-evaluate to any controllable level provides maximum flexibililty. However, no re-evalutaion is necessary in most common problems, and one re-evaluation is usually sufficient when re-evaluation is necessary.

It is also important to note that re-evaluation of a variable does not automatically assign the new value to the variable. For example, if we continue the above sequence by entering the expression

```
? A
```

then the resulting display is

```
B + 1.
```

In contrast, if we continue with the entry

```
? A = A@@;A
```

then the resulting display is

```
D + 8.
```

As another example illustrating the effects of one-level evaluation, suppose that after the input and response

```
? P = (U+1)^2
(U + 1)^2
```

we wish to expand P. It will not do to type <P> because expansion only applies to operations that are typed or re-evaluated while expansion is in effect. <P>@ won't work either, because re-evaluation must be done while expansion is on in order to expand P retrospectively. Thus, what we need is <P@>.

PARTIAL DERIVATIVES

The help message for partial derivatives is:

```
+---------------------------------------+
|To determine the partial derivative of |
|an expression with respect to a        |
|variable, enter an expression of the   |
|form                                   |
|                                       |
| expression % variable                 |
|                                       |
|% is names to suggest a ratio of       |
|infinitesimals.  % has the same        |
|precedence and left-to-right order as  |
|addition.  As examples try             |
|                                       |
|X='X; A='A; AX^3 + SIN X % X           |
|A^2 X^3 % X % A                        |
|                                       |
|After entering several such examples,  |
|enter a question mark to see the help  |
|menu.                                  |
|                                       |
|?_                                     |
+---------------------------------------+
```

These two inputs generate the corresponding outputs

3 A X^2 + COS X

6 A X^2.

As indicated by the second example, higher-order derivatives can be specified by repeated use of the operator.

As another example, to add 5 onto the derivative of log sin x, we could enter either

? LOG SIN X % X + 5

or

? 5 + (LOG SIN X % X)

Parentheses are necessary in the latter case because addition and differentiation have the same precedence and are done left to right.

Note that % is used somewhat differently in the APX ALGICALC program.

INTEGRALS

The help message screen for integrals is:

```
+------------------------------------------+
|To determine an antiderivative of an      |
|expression with respect to a variable,    |
|enter an expression of the form           |
|                                          |
| expression $ variable                    |
|                                          |
|$ is named to suggest an integral         |
|sign.  $ has the same precedence and      |
|left-to-right order as addition.  Avoid   |
|expansion of integrands, which may        |
|mask a known integrable pattern,          |
|because algebraic then harmonic           |
|expansion are tried automatically if      |
|the given form can't be integrated        |
|directly.  As examples, try               |
|                                          |
|X='X; A='A; 3AX^2 + COS X $ X             |
|COS LOG X / X $ X                         |
|                                          |
|After entering several such examples,     |
|enter a question mark to see the help     |
|menu.                                     |
|                                          |
|?_                                        |
+------------------------------------------+
```

These two examples produce the corresponding results:

```
A X^3 + SIN X

SIN LOG X
```

Note that as in most integral tables, a symbolic constant of integration is not included.

You may occasionally have to indicate the sign of some subexpressions. For example:

```
? 1/(x^2+c) $ x
Does sign (-4C) = +, - or 0?-
 ATN (C^-.5 X) C^-.5
```

Your response applies only to integration of one term, so you might have the same question more than once during a single integration problem. You can repeat the problem with alternative answers to determine their effect. In some cases different answers still lead to the same result.

The set of integration rules built into the program are curious. In these rules:

1. a, b, c, p and represent expressions that are independent of the variable x;

2. m and n represent integers with m>0 and n>1;

3. k represents any numeric constant other than -1.

4. f(x), g(x) and h(x) represent expressions contaning the variable x;

5. ' denotes partial differentiation with respect to x.

$\int$ a f(x) dx -> a $\int$ f(x) dx,

$\int$ (f(x) + g(x)) dx -> $\int$ f(x) dx + $\int$ g(x) dx,

$\int$ f(x) g(x) dx -> $f^2(x)/2$ if g(x) = f'(x),

$$\int f[h(x)]\, g(x)\, dx \;\to\; \frac{g(x)}{h'(x)} \int f[h(x)]\, dh(x) \quad \text{if } \frac{g(x)}{h'(x)} \text{ is free of } x,$$

$$\int (ax + b)^p\, dx \;\to\; \frac{(ax + b)^{p+1}}{a\,(p + 1)}$$

$\int (ax + b)^{-1}$ dx -> LOG (ax + b) / a,

$\int$ EXP (ax + b) dx -> EXP (ax + b) / a,

$\int$ COS (ax + b) dx -> SIN (ax + b) / a,

$\int$ SIN (ax + b) dx -> - COS (ax + b) / a,

$\int$ LOG (ax + b) dx -> (ax + b) LOG (ax + b) / a,

$\int$ TAN (ax + b) dx -> - LOG COS (ax + b) / a,

$\int$ COT (ax + b) dx -> LOG SIN (ax + b) / a,

$\int$ SEC (ax + b) dx -> LOG TAN [π/4 + (ax+b)/2] / a,

$\int$ CSC (ax + b) dx -> LOG TAN [(ax+b)/2] / a,

$\int$ ATN(ax + b) dx -> (ax+b) ATN(ax+b) / a - LOG[1+$(ax+b)^2$]/ (2a),

$\int$ ASN (ax + b) dx -> (ax+b) ASN(ax+b) / a + $[1-(ax+b)^2]^{.5}$ / a,

$\int$ ACS (ax + b) dx -> (ax+b) ACS(ax+b) / a - $[1-(ax+b)^2]^{.5}$ / a,

$\int$ $LOG^n x$ dx -> x $LOG^n x$ - n $\int$ $LOG^{n-1} x$ dx,

$\int SEC^n$(ax+b) dx -> [SIN(ax+b) SEC^{n-1}(ax+b) /a + (n-2)$\int SEC^{n-2}$(ax+b) dx]/(n-1),

$\int CSC^n$(ax+b)dx -> [-COS(ax+b)CSC^{n-1}(ax+b)/a + (n-2)$\int CSC^{n-2}$(ax+b)dx]/(n-1),

$\int$ TAN^n(ax+b) dx -> TAN^{n-1}(ax+b) / a / (n-1) - $\int$ TAN^{n-2}(ax+b) dx,

$\int$ COT^n(ax+b) dx -> $-COT^{n-1}$(ax+b) / a / (n-1) - $\int$ COT^{n-2}(ax+b) dx,

$\int$EXP(ax+b)SIN(px+q)dx -> EXP(ax+b) [a SIN(px+q) - p COS(px+q)] / (a^2+p^2),

$\int$EXP(ax+b)COS(px+q)dx -> EXP(ax+b) [a COS(px+q) + p SIN(px+q)] / (a^2+p^2),

$\int$ x^m EXP(ax+b) dx -> [x^m EXP(ax+b) - m $\int$ x^{m-1} EXP(ax+b) dx] / a,

$\int$ x^m SIN(ax+b) dx -> [m $\int$ x^{m-1} COS(ax+b) dx - x^m COS(ax+b)] / a,

$\int$ x^m COS(ax+b) dx -> [x^m SIN(ax+b) - m $\int$ x^{m-1} SIN(ax+b) dx] / a,

$\int$ x^m LOG x dx ->

$$\{x^{m+1} \text{ LOG } x - (-b/a)^{m+1} \sum_{k=1}^{m+1} (-ax/b)^k/k\} / (m+1),$$

$\int$ x^m $LOG^n x$ dx -> [x^{m+1} $LOG^n x$ - n $\int$ x^m $LOG^{n-1} x$ dx] / (m+1),

If r = ax^2 + bx + c, s = 2ax + b, and t = b^2 - 4ac, then:

$\int$ r^{-1} dx -> -2/s if t = 0,

-> $t^{-.5}$ LOG [$(t^{.5}-s)/(t^{.5}+s)$] if t > 0,

-> $(-t)^{-.5}$ ATN [s $(-t)^{-.5}$] if t < 0,

$\int$ $r^{-.5}$ dx -> $a^{-.5}$ LOG s if a = 0,

-> $a^{-.5}$ LOG [s + $(4ar)^{.5}$] if a > 0,

-> $(-a)^{-.5}$ ASN [s $t^{-.5}$] if a < 0,

$\int$ $r^{m-.5}$ dx -> [s r^2 - (m-.5) t $\int$ $r^{m-1.5}$ dx] / (4am),

$\int$ $r^{\wedge-n}$ dx - [2 a (3-2n) $\int$ r^{1-n} dx - s r^{1-n}] / t / (n-1).

To minimize the variety of operators within CALCULUS DEMON, negation is represented internally as multiplication by -1, subtraction is represented using addition and negation, reciprocals are represented using negative powers, and division is represented using multiplication and negative powers. Thus, the above rules also treat many instances of the operators "-" and "/".

If automatic transformation and the above rules do not fully integrate an expression, the expression is expanded algebraically, hoping that some of the transformation will convert the special terms into a form that agrees with the above rules. If there are still unintegrated terms, the expression is also harmonically expanded for a final attempt at integration. This set of rules covers as many as 50 percent to 90 percent of the examples in typical calculus texts.

The most significant classes of integrals for which the above rules are insufficient are those having nontrivial denominators or fractional powers. For example, the expression

```
? 1 / (X^3 + X^3 - X - 1) $ X
```

yields the response

```
(X^3 + X^2 - X - 1) ^ -1 $ X,
```

which is the DEMON's way of saying "you lose".

Equivalent expressions can differ in form, and alternative indefinite integrals also can differ by any constant. Consequently, CALCULUS DEMON will often yield indefinite integrals that differ from those in various tables. If you suspect that the CALCULUS DEMON is incorrect, differentiate it for comparison with the integrand. They may differ in form because of the various transformations done to aid integration. However, you can assign appropriate random constants to the variables, then re-evaluate the difference between the integrand and the derivative of its integral. This difference should be small compared to the coefficients therein unless several roundoff errors have occured. For example,

```
? G=X^2LOG(AX)^2; D=G$X%X
LOG A ^2 X^2 + LOG X ^2 X^2 + 2 LOG A LOG X X^2

? X=1.3; A=1.7; G@-D@
-2.98023E-08
```

Re-evaluation can be used to compute a definite integral in cases where it can be computed by substitution into an indefinite integral that CALCULUS DEMON can accommodate. For example, the following dialogue computes

$$\int_2^y cx^3\,dx:$$

```
? I= CX^3 $ X

[illegible]
```

```
? X=Y; U=I@; X=2; L=I@; X='X; J=U -L
 .25CY^4 - 4C
```

Moreover, this technique can be used to compute multiple definite integrals that can be done as iterated one-dimensional integrals. For example, to compute

start as above, then continue with: $\int_0^1 \left(\int_2^Y cx^3\,dx\right) dy$

```
? K=J$Y; Y=1; U=K@; Y=0;L=K@; Y='Y; U-L
-3.95C
```

HOW TO LEARN MORE ABOUT COMPUTER ALGEBRA

Many who use CALCULUS DEMON or another computer algebra system will want to learn more about this fascinating subject or will want to try other systems that have complementary or more powerful capabilities. Here is a brief guide to the relevant professional societies, the literature, and some widely available systems.

THE PROFESSIONAL SOCIETIES

The Association for Computing Machinery Special Interest Group on Symbolic and Algebraic Manipulation is the major international professional society for computer algebra. Their ACM SIGSAM Bulletin is the most concentrated and up-to-date source for abstracts and working papers together with announcements of meetings and systems. For information about joining, including special student rates, write the ACM at 1133 Avenue of the Americas, New York, NY 10036.

Some European groups devoted to computer algebra are:

1. SAM-AFCET: Contact M. Bergman, Faculte des Sciences de Luminy, Case 901, 13009; or contact J. Calmet, IMAG, B.P. 53, 38041 Grenoble Cedex, France.

2. NIGSAM: Contact Y. Sundblad, Department of Numerical Analysis and Computer Science, KTH S-10044 Stockholm, Sweden.

3. SEAS/SMC: Contact J. A. van Hulzen, Twente University of Technology, P.O. Box 217, 7500 AE enschede, The Netherlands.

THE LITERATURE

Regrettably, there is not yet a textbook devoted to computer algebra, and the few textbooks that contain relevant material are rather advanced. Most of the information is sparsely scattered in research journals or less accessible conference proceedings and reports. However, the following relatively accessible conference references contain surveys, bibliographies, and collections of articles that should serve as a good point of departure for exploring most facets of the literature:

1. ACM SIGSAM Bulletin, ACM, New York, all issues.

2. Communications of the ACM 14, No. 10, August 1971.

3. SIAM Journal on Computing 8, No. 3, August 1979.

4. Communications of the ACM 9, No. 10, August 1966.

5. Journal of the ACM 18, No. 4, October 1971.

6. P.S. Wang, editor, Proceedings of the 1981 ACM Symposium on Symbolic and Algebraic Computation, ACM Order No. 505810, P.O. Box 64145, Baltimore, MD 21264, $23.

7. E.W. Ng, editor, Symbolic and Algebraic Computation Lecture Notes in Computer Science, 72, Springer-Verlag, New York, 1979.

8. R.D. Jenks, editor, Proceedings of the 1976 ACM Symposium on Symbolic and Algebraic Computation, ACM, New York, 1976.

9. V.E. Lewis, editor, Proceedings of the 1979 MACSYMA User's Conference, M.I.T. Laboratory for Computer Science, 545 Technology Square, Cambridge, Massachusetts, 1979.

10. C.M. Anderson, editor, Proceedings of the 1977 MACSYMA User's Conference, NASA Cp-2012, 1977.

11. Knuth, D.E., The Art of Computer Programming, Volume II, Seminumerical Algorithms, Addison-Wesley, Reading, Mass., 1980.

12. Yun, D.Y.Y and Stoutemyer, D.R., "Symbolic Mathematical Computation", Encyclopedia of Computer Science and Technology, Suplememtary Volume 15, J. Belzer, A.G. Holzman and A. Kent, editors, M. Dekker, New York, pp. 235-310.

WIDELY AVAILABLE SYSTEMS

CALCULUS DEMON has two analogous companion programs that are distributed predominately for the same machines by the same software publishers.

ALGICALC treats expression that can simplify to ratios of polynomials in the unassigned variable X, and these expressions can be asssigned to any other variable A through Z. Relative to CALCULUS DEMON, this narrower class of expressions permits significantly faster and more powerful algebraic simplification. ALGICALC automatically reduces expressions over a common denominator, and the output can be displayed in either expanded or fully factored form. Regarding calculus operations, ALGICALC provides differentiation but not integration.

POLYCALC treats expressions that are polynomials in variables A through Z, generalized to permit negative and fractional powers of variables. Although CALCULUS DEMON is applicable to a larger class of expressions, POLYCALC is much faster, it can handle much larger problems without exhausting memory, and it includes a series truncation facility.

ALGICALC, POLYCALC and CALCULUS DEMON nicely complement each other.

Besides these, there are also larger systems that provide more capabilities at the expense of greater demands on memory space, computer sophistication, and cost. In contrast to the above programs:

1. They require anywhere up to 100 times as much memory space.

2. They may accommmodate a significantly larger class of expressions, including equations, vectors, matrices, or tensors.

3. They provide their own indefinite-precision arithmetic to avoid the serious

limitations of finite-precision arithmetic.

4. They accommodate much larger expressions having thousands of terms, with coefficients having hundreds of digits.

5. They provide a larger suite of built-in and optional transformations, including factoring and partial-fraction expansions.

6. They provide convenient facilities permitting the user to write function definitions and simplification rules to extend the built-in capabilities by enlarging the allowable class of expressions or the variety of available transformations. In other words, they are programmable.

In approximate order of increasing memory requirements, here are some of the most widely available general-purpose systems that are currently supported:

1. muMATH-80™ is an interactive system that runs on personal microcomputers based on the 6502, 8080, Z80, and 8086 or 8088 microprocessors, provided they have enough memory and an appropriate disk operating system. These include CP/M with at least 32 kilobytes of RAM, the Radio-Shack TRS-DOS with at least 32 kilobytes of such memory, the IBM PC with at least 64 kilobytes of such memory, or the Apple computer with at least 48 kilobytes of such memory. muMATH is distributed to end users, computer stores and hardware manufacturer by Microsoft at 10800 N.E. Eighth, Suite 819, Bellevue, Washington 98004, and muMATH is also distributed by the authors, The Soft Warehouse, at Box 11174, Honolulu, Hawaii 96828. The less common CP/M disk formats are available from Lifeboat Associates, 1651 Third Avenue, New York, N.Y. 10028.

2. SAC-2 is a non-interactive system which runs on any computer that can directly run a 1966 standard FORTRAN program of at least 120 kilobytes. Information about SAC-2 is available from Professor George Collins, Computer Sciences Department, University of Wisconsin, 1210 West Dayton St., Madison Wisconsin 53706.

3. FORMAC runs on any IBM 360 or 370 that can accommodate a PL/I program of at least 150 kilobytes. FORMAC is semi-interactive on some operating systems. Information about FORMAC is available from Knut Bahr at GMD/IFV, D-6100, Darmstadt, Germany.

4. ALTRAN is a non-interactive system which runs on any computer that can directly run a 1966 standard FORTRAN program of at least 270 kilobytes. Information about ALTRAN is available from the Computing Information Library, Bell Laboratories, 600 Mountain Avenue, Murray Hill, N.J. 07974.

5. REDUCE is an interactive system that runs on the IBM 360 or 370, DEC 10 or 20, Univac 1100 series, Contol Data Cyber series, Burroughs 6700, and several other computers, requiring a minimum of 350 kilobytes. For information about REDUCE, write Dr. Anthony Hearn, Rand Corporation, 1700 Main Street, Santa Monica, California 90401.

Additional systems are announced in back issues of the <u>ACM SIGSAM Bulletin</u>.

COMPUTER ALGEBRA IN EDUCATION

It should be clear to anyone who has experienced a computer algebra system that it has enormous potential for use in education as well as in research. Not only can computer algebra make computing more attractive to mathematically inclined students, computer algebra can make mathematics more attractive to computer enthusiasts. Computer algebra provides a great opportunity for mutual reinforcement, cross motivation, and computer education.

Personal computers are becoming so prevalent that students, engineers, scientists, and mathematicians will soon be using computer algebra extensively. Moreover, it should not be long before general-purpose computer algebra is available on pocket calculators, for the following reasons:

1. Several manufacuturers now make low wattage "CMOS" versions of most popular 8-bit microprocessors, and CMOS versions of some 16-bit microprocessors are currently under development.

2. There are already hand-held terminals with 32 kilobytes of low wattage read-write memory.

3. There are already hand-held calculators with a sufficiently large low wattage liquid-crystal to display a reasonably large mathematical expression -- perhaps one term at a time.

Eventually some enterprising manufacturer will merge these three technologies, producing a hand-held calculator capable of running a general-purpose computer-algebra system such as muMATH. It behooves every math and computer science educator to explore how this revolutionary tool can be used to aid education.

Most students are far more intrigued and motivated by the artificial intelligence and game playing applications of computers than by the accounting and numerical applications that currently account for most computer usage. It is advisable to exploit this strong preferential interest to help teach both mathematics and computer science. If more good math, science, and engineering students are attracted to computers and more good computer-oriented students are attracted to math, more students will ultimately learn to use computers effectively for both numeric and nonnumeric purposes.

Computer algebra makes a highly motivating introductory computer programming course for math, science and engineering students. Computer algebra is also an ideal principal language for such students, because numbers and arithmetic comprise appreciably less than half of the kindergarten through calculus math curriculum. Moreover, the limited-precision integer and floating-point arithmetic typical of traditional programming languages is not the kind of arithmetic taught in this curriculum or used in everyday life.

Some educators may fear that computer algebra might cause algebraic skills to atrophy or prevent them from ever developing. Analogous concerns were undoubtedly expressed about Arabic numerals, multiplication tables, logarithms, Laplace transforms, and numerical pocket calculators. We have survived their convenience. The National Council for Teachers of Mathematics strongly supports the use of numerical pocket calculators in

classrooms, and every reason for this support is even more true of computer algebra.

Automatic symbolic mathematics makes it possible for students to concentrate on basic mathematical concepts rather than spending an excessive amount of time mechanically performing transformations. Computer algebra lets students explore such fundamental concepts as commutativity, associativity, groups , rings, and fields. Moreover, the extensive algebraic capabilities of computer algebra enables students to investigate larger examples than are otherwise practical. Patterns revealed may suggest useful theorems. Conjectured patterns thus violated provide counterexamples against false hypotheses. Computer algebra can contribute to teaching mathematical discovery.

Existing computer-algebra systems could also make some of the following educational contributions:

1. Trace packages can be used to let students witness each step of an algebraic simplification, rather than merely the final result.

2. The very fact that algebra and calculus can be automated should encourage average and poor math students that the flashes of inspiration characteristic of quick students are unnecessary for those operations -- hope is revealed for the slower, more methodical students.

3. For students who know how to program in the language in which the computer algebra packages are written, inspection of the underlying algorithms may help them learn methods for accomplishing the operations. Moreover, by programming extensions to the built-in facilities, students can reinforce understanding of the built-in and extended operations.

Many existing computer algebra systems can be used in the above ways right now. However, there is a potential for much more. In conjunction with a computer-aided instruction package, existing computer algebra systems could be used for extremely flexible and intelligent algebra drill, testing, and tutorial dialogues.

None of the existing computer algebra systems is by itself a computer-aided math instruction system. However, someone experienced in computer-aided instruction could design interactive math lessons or tests to be used in conjunction with these existing computer algebra systems.

ALPHABETIZED ERROR MESSAGES: PROBABLE CAUSES

The CALCULUS DEMON error messages are detailed and precise, indicating exactly where processing could not continue and why. For example, if the error was caused by an ill-formed expression, then the program generally states every category of character that is allowable at that point. This section lists all of these messages in alphabetical order, along with some of the most common causes.

expected +, -, ', digit, decimal point, letter or (

> Perhaps an operator or a parenthesis occurs where it is not allowed, such as in the entries "X**2", ")", "()", or "*". Alternatively perhaps there is an illegal character such as "!", "#", or "&".

expected)

> CALCULUS DEMON encountered the end of an expression while seeking a right parenthesis to match an earlier left parenthesis.

expected different character

> There are extra characters that cannot be incorporated in the indicated complete expression. Common causes are excess right parentheses or illegal characters such as "!", "#", or "\".

expected existing topic number

> The expression preceding a question mark must evaluate to one of the integer lesson numbers listed in the help topic menu.

expected letter

> Only a letter is permitted after the apostrophe quote operator.

expected nonzero divisor

> An attempt was made to divide by a factor that evaluates to zero, and division by zero is mathematically undefined. This situation can arise indirectly when attempting to raise 0 to a negative power or when re-evaluation transforms a denominator subexpression into 0.

expected unassigned variable

> An attempt was made to differentiate or integrate with respect to a term that does not evaluate to the name of a variable. Perhaps the differentiation or integration variable has an assigned value that should be cleared by quoting.

Any other error messages are generated by Atari BASIC or DOS in a manner that was unanticipated by The Soft Warehouse. Refer to your DOS and BASIC Reference Manuals for assistance.

QUICK REFERENCE SHEET
FOR CALCULUS DEMON

<u>To enter problems</u>--> enter data [RETURN]
<u>To request HELP menu</u>--> ?[RETURN]

VARIABLE --> A-Z (capitals only)

ORDER OF OPERATIONS:

- ' (quoting)
- ^ (exponentiation)
- +,- (as signs)
- *,/ (multiplication, division)
- +,-,@,%,$ (addition, subtraction, re-evaluation, differentiation, integration)

MULTIPLE EXPONENTIATION: right to left, when no parentheses
MULTIPLE MULTIPLICATION AND DIVISION: left to right, when no parentheses
MULTIPLE ADDITION, SUBTRACTION, RE-EVALUATION, DIFFERENTIATION, INTEGRATION:
left to right when no parentheses

Use PARENTHESES IN DIVISION to include more than one factor as a denominator (e.g., 3/(xy))

Arithmetic limitations of ATARI BASIC
 Magnitude limitation for integers:
 LARGEST NONZERO MAGNITUDE: 10^96
 SMALLEST NONZERO MAGNITUDE: 10^-98

ASSIGNMENTS: letter=expression

To clear a variable: variable= 'variable

SUPPRESSED DISPLAY --> end input with ;
To enter MORE THAN ONE EXPRESSION/ASSIGNMENT PER LINE -->
separate them with ;

RE-EVALUATION: (sub)expression @

PARTIAL DERIVATIVES of expressions with respect to a variable:
(sub)expression % variable
Repeat % to express higher order derivatives

ANTIDERIVATIVES of expressions with respect to a variable:
(sub)expression $ variable

Limited Warranty on Media and Hardware Accessories. Atari, Inc. ("Atari") warrants to the original consumer purchaser that the media on which APX Computer Programs are recorded and any hardware accessories sold by APX shall be free from defects in material or workmanship for a period of thirty (30) days from the date of purchase. If you discover such a defect within the 30-day period, call APX for a return authorization number, and then return the product to APX along with proof of purchase date. We will repair or replace the product at our option. If you ship an APX product for in-warranty service, we suggest you package it securely with the problem indicated in writing and insure it for value, as Atari assumes no liability for loss or damage incurred during shipment.

This warranty shall not apply if the APX product has been damaged by accident, unreasonable use, use with any non-ATARI products, unauthorized service, or by other causes unrelated to defective materials or workmanship.

Any applicable implied warranties, including warranties of merchantability and fitness for a particular purpose, are also limited to thirty (30) days from the date of purchase. Consequential or incidental damages resulting from a breach of any applicable express or implied warranties are hereby excluded.

The provisions of the foregoing warranty are valid in the U.S. only. This warranty gives you specific legal rights and you may also have other rights which vary from state to state. Some states do not allow limitations on how long an implied warranty lasts, and/or do not allow the exclusion of incidental or consequential damages, so the above limitations and exclusions may not apply to you.

Disclaimer of Warranty on APX Computer Programs. Most APX Computer Programs have been written by people not employed by Atari. The programs we select for APX offer something of value that we want to make available to ATARI Home Computer owners. In order to economically offer these programs to the widest number of people, APX Computer Programs are not rigorously tested by Atari and are sold on an "as is" basis without warranty of any kind. Any statements concerning the capabilities or utility of APX Computer Programs are not to be construed as express or implied warranties.

Atari shall have no liability or responsibility to the original consumer purchaser or any other person or entity with respect to any claim, loss, liability, or damage caused or alleged to be caused directly or indirectly by APX Computer Programs. This disclaimer includes, but is not limited to, any interruption of services, loss of business or anticipatory profits, and/or incidental or consequential damages resulting from the purchase, use, or operation of APX Computer Programs.

Some states do not allow the limitation or exclusion of implied warranties or of incidental or consequential damages, so the above limitations or exclusions concerning APX Computer Programs may not apply to you.

ATARI®
PROGRAM
EXCHANGE
P.O. Box 3705
Santa Clara, CA 95055

Review Form

We're interested in your experiences with APX programs and documentation, both favorable and unfavorable. Many of our authors are eager to improve their programs if they know what you want. And, of course, we want to know about any bugs that slipped by us, so that the author can fix them. We also want to know whether our instructions are meeting your needs. You are our best source for suggesting improvements! Please help us by taking a moment to fill in this review sheet. Fold the sheet in thirds and seal it so that the address on the bottom of the back becomes the envelope front. Thank you for helping us!

1. Name and APX number of program.

____

____

2. If you have problems using the program, please describe them here.

____

____

____

3. What do you especially like about this program?

____

____

____

4. What do you think the program's weaknesses are?

____

____

____

5. How can the catalog description be more accurate or comprehensive?

____

____

6. On a scale of 1 to 10, 1 being "poor" and 10 being "excellent", please rate the following aspects of this program:

_______ Easy to use
_______ User-oriented (e.g., menus, prompts, clear language)
_______ Enjoyable
_______ Self-instructive
_______ Useful (non-game programs)
_______ Imaginative graphics and sound

7. Describe any technical errors you found in the user instructions (please give page numbers).

8. What did you especially like about the user instructions?

9 What revisions or additions would improve these instructions?

10. On a scale of 1 to 10, 1 representing "poor" and 10 representing "excellent", how would you rate the user instructions and why?

11. Other comments about the program or user instructions:

From

STAMP

ATARI Program Exchange
P.O. Box 3705
Santa Clara, CA 95055

[seal here]